Numer. Math. Theor. Meth. Appl.
doi: 10.4208/nmtma.2016.y15027

Vol. **9**, No. 4, pp. 549-578
November 2016

Bivariate Polynomial Interpolation over Nonrectangular Meshes

Jiang Qian[1,*], Sujuan Zheng[1], Fan Wang[2] and Zhuojia Fu[3]

[1] *College of Sciences, Hohai University, Nanjing 211000, China.*
[2] *College of Engineering, Nanjing Agricultural University, Nanjing 210031, China.*
[3] *College of Mechanics and Materials, Hohai University, Nanjing 211000, China.*

Received 4 November 2015; Accepted (in revised version) 16 March 2016

Abstract. In this paper, by means of a new recursive algorithm of non-tensor-product-typed divided differences, bivariate polynomial interpolation schemes are constructed over nonrectangular meshes firstly, which is converted into the study of scattered data interpolation. And the schemes are different as the number of scattered data is odd and even, respectively. Secondly, the corresponding error estimation is worked out, and an equivalence is obtained between high-order non-tensor-product-typed divided differences and high-order partial derivatives in the case of odd and even interpolating nodes, respectively. Thirdly, several numerical examples illustrate the recursive algorithms valid for the non-tensor-product-typed interpolating polynomials, and disclose that these polynomials change as the order of the interpolating nodes, although the node collection is invariant. Finally, from the aspect of computational complexity, the operation count with the bivariate polynomials presented is smaller than that with radial basis functions.

AMS subject classifications: 65D05

Key words: Bivariate polynomial interpolation; Nonrectangular meshes; Scattered data interpolation; The conformality of smoothing cofactor method; Radial basis function.

1. Introduction

The multivariate polynomial interpolation in the form of non-tensor product has many special features which do not arise in the simple univariate case directly. In fact, the geometry of the set of interpolation points (called nodes) is crucial for determining the solvability of the interpolation problem. Thus the challenging subject on multivariate interpolation has attracted much attention in the past. Salzer [14] pointed out that

*Corresponding author. *Email addresses:* qianjianghhu@sina.com (J. Qian), zsj@hhu.edu.cn (S.-J. Zheng), wangfan@njau.edu.cn (F. Wang), paul212063@hhu.edu.cn (Z.-J. Fu)

 ©2016 Global-Science Press

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

the existing interpolation formulae for more than one dimension possessed the desirability of at least three properties, i.e, truly irregular distribution of nodes, fairly simple recursion scheme for calculating successive coefficients, and limiting confluent forms. And he proposed a new bivariate divided difference algorithm to construct the bivariate polynomial interpolation over ortho-triples, the nodes of which were arranged in groups of three to form L-like configurations.

During many years, a vast amount work of multivariate polynomial interpolation has been done. Mazroui etc. [10] studied a recursive method for the construction of a Hermite spline interpolation. Madych [9] obtained an estimate of the $L^p(\Omega)$ norm of u in terms of the l^p norm of the values and its high-order derivatives, which is useful in obtaining error estimates for certain interpolation schemes. Dyn and Floater [5] studied multivariate polynomial interpolating on lower sets of points, which were expressed as the union of blocks of points. And Bailey [2] gave an equivalence between existence of particular exponential Riesz bases and existence of certain polynomial interpolants. Cui and Lei [4] solved one of the key problems in multivariate Birkhoff interpolation, which is to determining a monomial basis spanning the interpolation space. Allasia and Bracco [1] made a further study for Hermite-Birkhoff interpolation to multivariate real functions on scattered data by constructing a class of cardinal basis functions. And Chai etc. [3] proposed two algorithms to compute the basis of the minimal interpolation space and the lower interpolation space respectively.

However, to be mentioned, classical multivariate interpolation has always been restricted on the study of the case of tensor product [16], while multivariate interpolation in the form of non-tensor product has received not deep but constant attention [6] except scattered data interpolation with the well-known radial basis functions [21]. In the case of non-tensor-product-typed multivariate interpolation, on one hand, classic Bezout Theorem has played a crucial pole [16]. On the other hand, much attention has been paid on the study of convex preserving scattered data interpolation, which is due to multivariate splines [20]. Lai [7] used bivariate C^1 cubic splines to deal with convexity preserving scattered data interpolation problem. And to improve the order of approximation, Zhou and Lai [22] proposed several new schemes. Zhu and Wang [23] presented a method to construct Lagrange interpolation sets for bivariate spline spaces on cross-cut partitions by using interpolation along a piecewise algebraic curve. The above construction of the bivariate polynomial interpolation schemes needs the spline bases in the minimal supports, which can be worked out by means of the Conformality of Smoothing Cofactor Method. For the computation of the spline bases, one can see [8, 11–13, 17, 20].

Being Inspired by the construction of bivariate polynomial interpolation scheme over ortho-triples in [14], we have considered the case of bivariate continued fraction interpolation over the ortho-triples [18, 19], which is based on a new partial inverse divided differences. However, to the best of our knowledge, there are few papers on the study of bivariate interpolation over general triples or nonrectangular mesh. Thus with inspiration of Salze's thoughts [14], we have continued the study of the subject and have obtained achievements, some of which are shown in this paper.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

A brief outline of this article is as the following. Firstly, in Section 2, we propose new recursive algorithms of non-tensor-product-typed bivariate divided differences, and determine the interpolating coefficients. Secondly, in Section 3, we work out error estimation of the bivariate polynomial interpolation over nonrectangular meshes from the node collection with odd numbers, and obtain an equivalence between new non-tensor-product-type high-order divided differences and high-order partial derivatives. Moreover, in Section 4, we obtain error estimation of the bivariate polynomial interpolation over nonrectangular meshes from the node collection with even numbers, which is similar but different with the case in Section 3. Thirdly, in Section 5, we compute specific interpolating polynomials with different degrees in some numerical examples. Finally, Section 6 analyzes the computational complexity of operations with the bivariate interpolating polynomials presented and the well-known radial basis functions.

2. Interpolating coefficients based on recursive algorithms

In this section, we shall propose new recursive algorithms of non-tensor-product-typed bivariate divided differences, and construct interpolating polynomial schemes over nonrectangular meshes. Then it can be converted into scattered data interpolation over node collection.

Firstly, suppose that the $2n+1$ distinct points are arranged in $\Omega_{2n} = \{(x_0, y_0), (x_1, y_1), \cdots, (x_{2n}, y_{2n})\}$, which is called the interpolating node collection. And we choose the interpolating nodes to guarantee the smooth computation in the paper, for instance, $x_i \neq x_j$, $y_i \neq y_j$, for $i \neq j$. We shall consider the interpolating bivariate polynomial as the following type of representation

$$\begin{aligned} p_{2n}(x,y) = {} & a_0 + a_1(x-x_0) + a_2(y-y_0)(x-x_1) + \cdots \\ & + a_{2n-1}(x-x_0)(y-y_1)(x-x_2)(y-y_3)\cdots(x-x_{2n-2}) \\ & + a_{2n}(y-y_0)(x-x_1)(y-y_2)(x-x_3)\cdots(y-y_{2n-2})(x-x_{2n-1}). \end{aligned} \tag{2.1}$$

Secondly, suppose that the $2n+2$ distinct points are arranged in $\Omega_{2n+1} = \{(x_0, y_0), (x_1, y_1), \cdots, (x_{2n+1}, y_{2n+1})\}$, where $x_i \neq x_j$, $y_i \neq y_j$, for $i \neq j$. We shall consider the interpolating bivariate polynomial as the following representation

$$p_{2n+1}(x,y) = p_{2n}(x,y) + a_{2n+1}(x-x_0)(y-y_1)\cdots(x-x_{2n}), \tag{2.2}$$

where $p_{2n}(x,y)$ is defined as (2.1).

In order to put (2.1)and (2.2) into application in bivariate polynomial interpolation over Ω_{2n} and Ω_{2n+1}, respectively, we shall develop a new non-tensor-product-typed algorithm of bivariate divided differences. And the interpolating coefficients can be worked out with the algorithm. To save space, the bivariate divided difference $[x_0, \cdots, x_k;\ y_0, \cdots, y_k]$, i.e., $F_{0,\cdots,k}$ may be used to denote $[x_0, x_1, \cdots, x_{k-1}, x_k;\ y_0, y_1, \cdots, y_{k-1}, y_k]$, and additional letters may be written explicitly only when the subscripts are not consecutive. And we denote the vertical coordinates at the interpolating

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

nodes from Ω_{2n} as $z_i = f(x_i, y_i) \equiv f_{i,i}$ for $i = 0, 1, \cdots, 2n, 2n+1$, where $f(x, y)$ is the interpolated function.

Definition 2.1 *Let the interpolating node collection Ω_{2n}, where $x_i \neq x_j, y_i \neq y_j$, for $i \neq j$, we define the following bivariate non-tensor-product-typed divided differences.*

(1).

$$F_i \equiv z_i = f(x_i, y_i) = f_{i,i} = [x_i;\ y_i], \quad i = 0, 1, \cdots, 2n, 2n+1. \tag{2.3}$$

(2).

$$F_{0,1} \equiv [x_0, x_1;\ y_0, y_1] = \frac{f_{1,1} - f_{0,0}}{x_1 - x_0}. \tag{2.4}$$

(3).

$$\begin{aligned} F_{0,1,2} &\equiv [x_0, x_1, x_2;\ y_0, y_1, y_2] \\ &= \frac{[x_0, x_2;\ y_0, y_2]_y - [x_0, x_1;\ y_0, y_1]\dfrac{x_2 - x_0}{y_2 - y_0}}{x_2 - x_1}, \end{aligned} \tag{2.5}$$

where $[x_0, x_1;\ y_0, y_1]$ is defined as (2.4)*, and*

$$F_{0,2,y} \equiv [x_0, x_2;\ y_0, y_2]_y = \frac{f_{2,2} - f_{0,0}}{y_2 - y_0}.$$

(4).

$$\begin{aligned} F_{0,\cdots,3} &\equiv [x_0, \cdots, x_3;\ y_0, \cdots, y_3] \\ &= \frac{[x_0, x_1, x_3;\ y_0, y_1, y_3]_y - [x_0, x_1, x_2;\ y_0, y_1, y_2]\dfrac{(y_3 - y_0)(x_3 - x_1)}{(x_3 - x_0)(y_3 - y_1)}}{x_3 - x_2}, \end{aligned} \tag{2.6}$$

where $[x_0, x_1, x_2;\ y_0, y_1, y_2]$ is defined as (2.5)*, and*

$$F_{0,1,3,y} \equiv [x_0, x_1, x_3;\ y_0, y_1, y_3]_y = \frac{[x_0, x_3;\ y_0, y_3] - [x_0, x_1;\ y_0, y_1]}{y_3 - y_1},$$

$$F_{0,3} \equiv [x_0, x_3;\ y_0, y_3] = \frac{f_{3,3} - f_{0,0}}{x_3 - x_0}.$$

(5).

$$\begin{aligned} F_{0,\cdots,4} &\equiv [x_0, \cdots, x_4;\ y_0, \cdots, y_4] \\ &= \Big\{ [x_0, x_1, x_2, x_4;\ y_0, y_1, y_2, y_4]_y - [x_0, \cdots, x_3;\ y_0, \cdots, y_3] \\ &\quad \cdot \frac{(x_4 - x_0)(y_4 - y_1)(x_4 - x_2)}{(y_4 - y_0)(x_4 - x_1)(y_4 - y_2)} \Big\} \Big/ (x_4 - x_3), \end{aligned} \tag{2.7}$$

where $[x_0, \cdots, x_3;\ y_0, \cdots, y_3]$ is defined as (2.6)*, and*

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

$$F_{0,1,2,4,y} \equiv [x_0, x_1, x_2, x_4;\ y_0, y_1, y_2, y_4]_y$$
$$= \frac{[x_0, x_1, x_4;\ y_0, y_1, y_4] - [x_0, x_1, x_2;\ y_0, y_1, y_2]}{y_4 - y_2},$$
$$F_{0,1,4} \equiv [x_0, x_1, x_4;\ y_0, y_1, y_4] = \frac{[x_0, x_4;\ y_0, y_4]_y - [x_0, x_1;\ y_0, y_1]\dfrac{x_4 - x_0}{y_4 - y_0}}{x_4 - x_1},$$
$$F_{0,4,y} \equiv [x_0, x_4;\ y_0, y_4]_y = \frac{f_{4,4} - f_{0,0}}{y_4 - y_0}.$$
$$\cdots$$

(2n+1).

$$F_{0,\cdots,2n} \equiv [x_0, \cdots, x_{2n};\ y_0, \cdots, y_{2n}] \tag{2.8}$$
$$= \Big\{ [x_0, \cdots, x_{2n-2}, x_{2n};\ y_0, \cdots, y_{2n-2}, y_{2n}]_y - [x_0, \cdots, x_{2n-1};\ y_0, \cdots, y_{2n-1}]$$
$$\cdot \frac{(x_{2n} - x_0)(y_{2n} - y_1)\cdots(x_{2n} - x_{2n-2})}{(y_{2n} - y_0)(x_{2n} - x_1)\cdots(y_{2n} - y_{2n-2})} \Big\} \Big/ (x_{2n} - x_{2n-1}),$$

where

$$F_{0,\cdots,2n-2,2n,y} \equiv [x_0, \cdots, x_{2n-2}, x_{2n};\ y_0, \cdots, y_{2n-2}, y_{2n}]_y$$
$$= \frac{[x_0, \cdots, x_{2n-3}, x_{2n};\ y_0, \cdots, y_{2n-3}, y_{2n}] - [x_0, \cdots, x_{2n-2};\ y_0, \cdots, y_{2n-2}]}{y_{2n} - y_{2n-2}},$$
$$\cdots$$
$$F_{0,\cdots,5,2n} \equiv [x_0, \cdots, x_5, x_{2n};\ y_0, \cdots, y_5, y_{2n}]$$
$$= \Big\{ [x_0, \cdots, x_4, x_{2n};\ y_0, \cdots, y_4, y_{2n}]_y - [x_0, \cdots, x_5;\ y_0, \cdots, y_5]$$
$$\cdot \frac{(x_{2n} - x_0)(y_{2n} - y_1)\cdots(x_{2n} - x_4)}{(y_{2n} - y_0)(x_{2n} - x_1)\cdots(y_{2n} - y_4)} \Big\} \Big/ (x_{2n} - x_5),$$
$$F_{0,\cdots,4,2n,y} \equiv [x_0, \cdots, x_4, x_{2n};\ y_0, \cdots, y_4, y_{2n}]_y$$
$$= \frac{[x_0, \cdots, x_3, x_{2n};\ y_0, \cdots, y_3, y_{2n}] - [x_0, \cdots, x_4;\ y_0, \cdots, y_4]}{y_{2n} - y_4},$$
$$F_{0,\cdots,3,2n} \equiv [x_0, \cdots, x_3, x_{2n};\ y_0, \cdots, y_3, y_{2n}]$$
$$= \Big\{ [x_0, x_1, x_2, x_{2n};\ y_0, y_1, y_2, y_{2n}]_y - [x_0, \cdots, x_3;\ y_0, \cdots, y_3]$$
$$\cdot \frac{(x_{2n} - x_0)(y_{2n} - y_1)(x_{2n} - x_2)}{(y_{2n} - y_0)(x_{2n} - x_1)(y_{2n} - y_2)} \Big\} \Big/ (x_{2n} - x_3),$$
$$F_{0,1,2,2n,y} \equiv [x_0, x_1, x_2, x_{2n};\ y_0, y_1, y_2, y_{2n}]_y$$
$$= \frac{[x_0, x_1, x_{2n};\ y_0, y_1, y_{2n}] - [x_0, x_1, x_2;\ y_0, y_1, y_2]}{y_{2n} - y_2},$$
$$F_{0,1,2n} \equiv [x_0, x_1, x_{2n};\ y_0, y_1, y_{2n}] = \frac{[x_0, x_{2n};\ y_0, y_{2n}]_y - [x_0, x_1;\ y_0, y_1]\dfrac{x_{2n} - x_0}{y_{2n} - y_0}}{x_{2n} - x_1},$$
$$F_{0,2n,y} \equiv [x_0, x_{2n};\ y_0, y_{2n}]_y = \frac{[x_{2n};\ y_{2n}] - [x_0;\ y_0]}{y_{2n} - y_0}.$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

(2n+2).

$$
\begin{aligned}
F_{0,\cdots,2n+1} \equiv & [x_0,\cdots,x_{2n+1};\ y_0,\cdots,y_{2n+1}] \qquad (2.9)\\
= & \Big\{ [x_0,\cdots,x_{2n-1},x_{2n+1};\ y_0,\cdots,y_{2n-1},y_{2n+1}]_y - [x_0,\cdots,x_{2n};\ y_0,\cdots,y_{2n}] \\
& \cdot \frac{(y_{2n+1}-y_0)\cdots(x_{2n+1}-x_{2n-1})}{(x_{2n+1}-x_0)\cdots(y_{2n+1}-y_{2n-1})} \Big\} \Big/ \big(x_{2n+1}-x_{2n}\big),
\end{aligned}
$$

where

$$
\begin{aligned}
& F_{0,\cdots,2n-1,2n+1,y} \equiv [x_0,\cdots,x_{2n-1},x_{2n+1};\ y_0,\cdots,y_{2n-1},y_{2n+1}]_y \\
& = \frac{[x_0,\cdots,x_{2n-2},x_{2n+1};\ y_0,\cdots,y_{2n-2},y_{2n+1}] - [x_0,\cdots,x_{2n-1};\ y_0,\cdots,y_{2n-1}]}{y_{2n+1}-y_{2n-1}}, \\
& \qquad \cdots \\
& F_{0,\cdots,4,2n+1} \equiv [x_0,\cdots,x_4,x_{2n+1};\ y_0,\cdots,y_4,y_{2n+1}] \\
& = \Big\{ [x_0,\cdots,x_3,x_{2n+1};\ y_0,\cdots,y_3,y_{2n+1}]_y - [x_0,\cdots,x_4;\ y_0,\cdots,y_4] \\
& \quad \cdot \frac{(y_{2n+1}-y_0)\cdots(x_{2n+1}-x_3)}{(x_{2n+1}-x_0)\cdots(y_{2n+1}-y_3)} \Big\} \Big/ \big(x_{2n+1}-x_4\big), \\
& F_{0,\cdots,3,2n+1,y} \equiv [x_0,\cdots,x_3,x_{2n+1};\ y_0,\cdots,y_3,y_{2n+1}]_y \\
& = \frac{[x_0,x_1,x_2,x_{2n+1};\ y_0,y_1,y_2,y_{2n+1}] - [x_0,\cdots,x_3;\ y_0,\cdots,y_3]}{y_{2n+1}-y_3}, \\
& F_{0,1,2,2n+1} \equiv [x_0,x_1,x_2,x_{2n+1};\ y_0,y_1,y_2,y_{2n+1}] \\
& = \Big\{ [x_0,x_1,x_{2n+1};\ y_0,y_1,y_{2n+1}]_y - [x_0,x_1,x_2;\ y_0,y_1,y_2] \\
& \quad \cdot \frac{(y_{2n+1}-y_0)(x_{2n+1}-x_1)}{(x_{2n+1}-x_0)(y_{2n+1}-y_1)} \Big\} \Big/ \big(x_{2n+1}-x_2\big), \\
& F_{0,1,2n+1,y} \equiv [x_0,x_1,x_{2n+1};\ y_0,y_1,y_{2n+1}]_y = \frac{[x_0,x_{2n+1};\ y_0,y_{2n+1}] - [x_0,x_1;\ y_0,y_1]}{y_{2n+1}-y_1}, \\
& F_{0,2n+1} \equiv [x_0,x_{2n+1};\ y_0,y_{2n+1}] = \frac{[x_{2n+1};\ y_{2n+1}] - [x_0;\ y_0]}{x_{2n+1}-x_0}.
\end{aligned}
$$

We summarize the results in Definition 2.1 into the corresponding algorithms for the computation of the bivariate non-tensor-product-typed divided differences with odd and even interpolating nodes, respectively, which are also shown in Table 1.

Remark 2.1. In order to guarantee the smooth computation in Algorithm 2.1 and 2.2, one can choose the interpolating nodes which lie on a diagonal line in rectangular mesh as an example.

Based upon Algorithm 2.1 and 2.2, finite-sum identities can be obtained where the interpolating coefficients are represented in bivariate non-tensor-product-typed divided differences and the last term is a function of x and y.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

Algorithm 2.1

1. *Initialization:* $F_i = f(x_i, y_i)$, *where* $i = 0, 1, \cdots, 2n$.
2. *Recursive case: By means of Definition* 2.1, *calculate*

$$\begin{aligned} & F_{0,i}(i = 1, 3, \cdots, 2n-1), \quad F_{0,j,y}(j = 2, 4, \cdots, 2n) \\ \rightarrow \quad & F_{0,1,j}(j = 2, 4, \cdots, 2n), \quad F_{0,1,i,y}(i = 3, 5, \cdots, 2n-1) \\ \rightarrow \quad & F_{0,1,2,i}(i = 3, 5, \cdots, 2n), \quad F_{0,1,2,j,y}(j = 4, 6, \cdots, 2n) \\ \rightarrow \quad & F_{0,\cdots,3,j}(j = 4, 6, \cdots, 2n), \quad F_{0,\cdots,3,i,y}(i = 5, 7, \cdots, 2n-1) \rightarrow \cdots \\ \rightarrow \quad & F_{0,\cdots,2n-1}, \quad F_{0,\cdots,2n-2,2n,y}. \end{aligned}$$

3. *Result:* $F_{0\cdots,2n}$.

Algorithm 2.2

1. *Initialization:* $F_i = f(x_i, y_i)$, *where* $i = 0, 1, \cdots, 2n+1$.
2. *Recursive case: By means of Definition* 2.1, *calculate*

$$\begin{aligned} & F_{0,i}(i = 1, 3, \cdots, 2n-1), \quad F_{0,j,y}(j = 2, 4, \cdots, 2n) \\ \rightarrow \quad & F_{0,1,j}(j = 2, 4, \cdots, 2n), \quad F_{0,1,i,y}(i = 3, 5, \cdots, 2n-1) \\ \rightarrow \quad & F_{0,1,2,i}(i = 3, 5, \cdots, 2n), \quad F_{0,1,2,j,y}(j = 4, 6, \cdots, 2n) \\ \rightarrow \quad & F_{0,\cdots,3,j}(j = 4, 6, \cdots, 2n), \quad F_{0,\cdots,3,i,y}(i = 5, 7, \cdots, 2n-1) \rightarrow \cdots \\ \rightarrow \quad & F_{0,\cdots,2n}, \quad F_{0,\cdots,2n-1,2n+1,y}. \end{aligned}$$

3. *Result:* $F_{0\cdots,2n+1}$.

Table 1: Recursive computation of bivariate non-tensor-product-typed divided differences.

F_0	F_1	F_2	F_3	F_4	F_5	$\cdots$	F_{2n}	F_{2n+1}
	$F_{0,1}$	$F_{0,2,y}$	$F_{0,3}$	$F_{0,4,y}$	$F_{0,5}$	$\cdots$	$F_{0,2n,y}$	$F_{0,2n+1}$
		$F_{0,1,2}$	$F_{0,1,3,y}$	$F_{0,1,4}$	$F_{0,1,5,y}$	$\cdots$	$F_{0,1,2n}$	$F_{0,1,2n+1,y}$
			$F_{0,\cdots,3}$	$F_{0,1,2,4,y}$	$F_{0,1,2,5}$	$\cdots$	$F_{0,1,2,2n,y}$	$F_{0,1,2,2n+1}$
				$F_{0,\cdots 4}$	$F_{0,\cdots,3,5,y}$	$\cdots$	$F_{0,\cdots,3,2n}$	$F_{0,\cdots,3,2n+1,y}$
					$F_{0,\cdots,5}$	$\cdots$	$F_{0,\cdots,4,2n,y}$	$F_{0,\cdots,4,2n+1}$
						$\ddots$	$\vdots$	$\vdots$
							$F_{0,\cdots,2n}$	$F_{0,\cdots,2n-1,2n+1,y}$
								$F_{0,\cdots,2n+1}$

Theorem 2.1. *For all interpolating nodes in* Ω_{2n},

$$\begin{aligned} f(x, y) \equiv & \, p_{2n}(x, y) \\ & + a_{2n+1}(x, y)(x - x_0)(y - y_1)(x - x_2) \cdots (y - y_{2n-1})(x - x_{2n}), \end{aligned} \tag{2.10}$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

where the bivariate polynomial $p_{2n}(x,y)$ is defined as (2.1). *And the corresponding coefficients can be worked out with Algorithm* 2.1 *and* 2.2*,*

$$\begin{cases} a_{2i} = [x_0, \cdots, x_{2i};\ y_0, \cdots, y_{2i}], & i = 0, 1, \cdots, n, \\ a_{2i-1} = [x_0, \cdots, x_{2i-1};\ y_0, \cdots, y_{2i-1}], & i = 1, \cdots, n, \end{cases} \tag{2.11}$$

and

$$\begin{aligned} a_{2n+1}(x,y) &= [x_0, \cdots, x_{2n}, x;\ y_0, \cdots, y_{2n}, y] \\ &= \Big\{ [x_0, \cdots, x_{2n-1}, x;\ y_0, \cdots, y_{2n-1}, y]_y - [x_0, \cdots, x_{2n};\ y_0, \cdots, y_{2n}] \\ &\quad \cdot \frac{(y-y_0)(x-x_1)\cdots(y-y_{2n-2})(x-x_{2n-1})}{(x-x_0)(y-y_1)\cdots(x-x_{2n-2})(y-y_{2n-1})} \Big\} \Big/ (x-x_{2n}), \end{aligned} \tag{2.12}$$

where

$$\begin{aligned} &[x_0, \cdots, x_{2n-1}, x;\ y_0, \cdots, y_{2n-1}, y]_y \\ &= \frac{[x_0, \cdots, x_{2n-2}, x;\ y_0, \cdots, y_{2n-2}, y] - [x_0, \cdots, x_{2n-1};\ y_0, \cdots, y_{2n-1}]}{y - y_{2n-1}}, \\ &\qquad \cdots \\ &[x_0, \cdots, x_4, x;\ y_0, \cdots, y_4, y] \\ &= \Big\{ [x_0, \cdots, x_3, x;\ y_0, \cdots, y_3, y]_y - [x_0, \cdots, x_4;\ y_0, \cdots, y_4] \\ &\qquad \cdot \frac{(y-y_0)\cdots(x-x_3)}{(x-x_0)\cdots(y-y_3)} \Big\} \Big/ (x-x_4), \\ &[x_0, \cdots, x_3, x;\ y_0, \cdots, y_3, y]_y \\ &= \frac{[x_0, x_1, x_2, x;\ y_0, y_1, y_2, y] - [x_0, \cdots, x_3;\ y_0, \cdots, y_3]}{y_{2n+1} - y_3}, \\ &[x_0, x_1, x_2, x;\ y_0, y_1, y_2, y] \\ &= \frac{[x_0, x_1, x;\ y_0, y_1, y]_y - [x_0, x_1, x_2;\ y_0, y_1, y_2]\dfrac{(y-y_0)(x-x_1)}{(x-x_0)(y-y_1)}}{x - x_2}, \\ &[x_0, x_1, x;\ y_0, y_1, y]_y = \frac{[x_0, x;\ y_0, y] - [x_0, x_1;\ y_0, y_1]}{y - y_1}, \\ &[x_0, x;\ y_0, y] = \frac{[x;\ y] - [x_0;\ y_0]}{x - x_0}, \\ &[x;\ y] = f(x,y). \end{aligned}$$

Theorem 2.2. *For Ω_{2n+1},*

$$\begin{aligned} f(x,y) \equiv\ & p_{2n+1}(x,y) \\ & + a_{2n+2}(x,y)(y-y_0)(x-x_1)\cdots(y-y_{2n})(x-x_{2n+1}), \end{aligned} \tag{2.13}$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

where the bivariate polynomial $p_{2n+1}(x,y)$ *is defined as* (2.2). *And for* $i=0,1,\cdots,n$, *the corresponding coefficients can be worked out with Algorithm* 2.1 *and* 2.2,

$$\begin{cases} a_{2i}=[x_0,\cdots,x_{2i};\ y_0,\cdots,y_{2i}], \\ a_{2i+1}=[x_0,\cdots,x_{2i-1};\ y_0,\cdots,y_{2i+1}], \end{cases} \tag{2.14}$$

and

$$\begin{aligned} a_{2n+2}(x,y) &= [x_0,\cdots,x_{2n+1},x;\ y_0,\cdots,y_{2n+1},y] \\ &= \Big\{[x_0,\cdots,x_{2n},x;\ y_0,\cdots,y_{2n},y]_y - [x_0,\cdots,x_{2n+1};\ y_0,\cdots,y_{2n+1}] \\ &\quad \cdot \frac{(x-x_0)(y-y_1)\cdots(x-x_{2n})}{(y-y_0)(x-x_1)\cdots(y-y_{2n})}\Big\}\Big/(x-x_{2n+1}), \end{aligned} \tag{2.15}$$

where

$$\begin{aligned} &[x_0,\cdots,x_{2n},x;\ y_0,\cdots,y_{2n},y]_y \\ &= \frac{[x_0,\cdots,x_{2n-1},x;\ y_0,\cdots,y_{2n-1},y]-[x_0,\cdots,x_{2n};\ y_0,\cdots,y_{2n}]}{y-y_{2n}}, \\ &\quad \cdots \\ &[x_0,\cdots,x_5,x;\ y_0,\cdots,y_5,y] \\ &= \Big\{[x_0,\cdots,x_4,x;\ y_0,\cdots,y_4,y]_y - [x_0,\cdots,x_5;\ y_0,\cdots,y_5] \\ &\quad \cdot \frac{(x-x_0)(y-y_1)\cdots(x-x_4)}{(y-y_0)(x-x_1)\cdots(y-y_4)}\Big\}\Big/(x-x_5), \\ &[x_0,\cdots,x_4,x;\ y_0,\cdots,y_4,y]_y \\ &= \frac{[x_0,\cdots,x_3,x;\ y_0,\cdots,y_3,y]-[x_0,\cdots,x_4;\ y_0,\cdots,y_4]}{y-y_4}, \\ &[x_0,\cdots,x_3,x;\ y_0,\cdots,y_3,y] \\ &= \Big\{[x_0,x_1,x_2,x;\ y_0,y_1,y_2,y]_y - [x_0,\cdots,x_3;\ y_0,\cdots,y_3] \\ &\quad \cdot \frac{(x-x_0)(y-y_1)(x-x_2)}{(y-y_0)(x-x_1)(y-y_2)}\Big\}\Big/(x-x_3), \\ &[x_0,x_1,x_2,x;\ y_0,y_1,y_2,y]_y = \frac{[x_0,x_1,x;\ y_0,y_1,y]-[x_0,x_1,x_2;\ y_0,y_1,y_2]}{y-y_2}, \\ &[x_0,x_1,x;\ y_0,y_1,y] = \frac{[x_0,x;\ y_0,y]_y - [x_0,x_1;\ y_0,y_1]\dfrac{x-x_0}{y-y_0}}{x-x_1}, \\ &[x_0,x;\ y_0,y]_y = \frac{[x;\ y]-[x_0;\ y_0]}{y-y_0}, \\ &[x;\ y]=f(x,y). \end{aligned}$$

Proof of Theorems 2.1 and 2.2. The proof of Theorems 2.1 and 2.2 is performed for n by induction at the same time.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

For $n = 0,\ \Omega_0 = \{(x_0, y_0)\}$,

$$f(x, y) \equiv p_0(x, y) + a_1(x, y)(x - x_0) = a_0 + a_1(x, y)(x - x_0).$$

Thus, by using the interpolation over Ω_0, we have

$$\begin{aligned} a_0 &= f(x_0, y_0) = f_{0,0}, \\ a_1(x, y) &= \frac{f(x, y) - f_{0,0}}{x - x_0} = [x_0, x;\ y_0, y]. \end{aligned}$$

For $n = 1,\ \Omega_1 = \{(x_0, y_0), (x_1, y_1)\}$,

$$\begin{aligned} f(x, y) &\equiv p_1(x, y) + a_2(x, y)(y - y_0)(x - x_1) \\ &= a_0 + a_1(x - x_0) + a_2(x, y)(y - y_0)(x - x_1). \end{aligned}$$

Thus, based on the interpolation over Ω_1, it follows

$$\begin{aligned} a_1 &= \frac{f_{1,1} - f_{0,0}}{x - x_0} = [x_0, x_1;\ y_0, y_1], \\ a_2(x, y) &= \frac{f(x, y) - a_0 - a_1(x - x_0)}{(y - y_0)(x - x_1)} \\ &= \frac{\dfrac{f(x, y) - f_{0,0}}{y_2 - y_0} - [x_0, x_1;\ y_0, y_1]\dfrac{x - x_0}{y - y_0}}{x - x_1} \\ &= \frac{[x_0, x;\ y_0, y]_y - [x_0, x_1;\ y_0, y_1]\dfrac{x - x_0}{y - y_0}}{x - x_1} \\ &= [x_0, x_1, x;\ y_0, y_1, y]. \end{aligned}$$

For $n = 2,\ \Omega_2 = \{(x_i, y_i), i = 0, 1, 2\}$,

$$\begin{aligned} f(x, y) &\equiv p_2(x, y) + a_3(x, y)(x - x_0)(y - y_1)(x - x_2) \\ &= a_0 + a_1(x - x_0) + a_2(y - y_0)(x - x_1) \\ &\quad + a_3(x, y)(x - x_0)(y - y_1)(x - x_2). \end{aligned}$$

Thus, by means of the interpolation over Ω_2, it follows from the above

$$\begin{aligned} a_2 &= \frac{f(x_2, y_2) - a_0 - a_1(x - x_0)}{(y - y_0)(x - x_1)} \\ &= \frac{\dfrac{f_{2,2} - f_{0,0}}{y_2 - y_0} - [x_0, x_1;\ y_0, y_1]\dfrac{x_2 - x_0}{y_2 - y_0}}{x_2 - x_1} \\ &= \frac{[x_0, x_2;\ y_0, y_2]_y - [x_0, x_1;\ y_0, y_1]\dfrac{x_2 - x_0}{y_2 - y_0}}{x_2 - x_1} \\ &= [x_0, x_1, x_2;\ y_0, y_1, y_2], \end{aligned}$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

$$
\begin{aligned}
a_3(x,y) &= \frac{f(x,y)-a_0-a_1(x-x_0)-a_2(y-y_0)(x-x_1)}{(x-x_0)(y-y_1)(x-x_2)}\\
&= \frac{\dfrac{f(x,y)-f_{0,0}}{x-x_0}-[x_0,x_1;\ y_0,y_1]}{y-y_1}-[x_0,x_1,x_2;\ y_0,y_1,y_2]\frac{(y-y_0)(x-x_1)}{(x-x_0)(y-y_1)}\\
&= \left\{\frac{[x_0,x_1,x;\ y_0,y_1,y]_y-[x_0,x_1;\ y_0,y_1]}{y-y_1}-[x_0,x_1,x_2;\ y_0,y_1,y_2]\right.\\
&\quad \left.\cdot\frac{(y-y_0)(x-x_1)}{(x-x_0)(y-y_1)}\right\}\Big/(x-x_2)\\
&= [x_0,x_1,x_2,x;\ y_0,y_1,y_2,x].
\end{aligned}
$$

For $n=3,\ \Omega_3=\{(x_i,y_i), i=0,\cdots,3\}$,

$$
\begin{aligned}
f(x,y) &\equiv p_3(x,y)+a_4(x,y)(y-y_0)(x-x_1)(y-y_2)(x-x_3)\\
&= a_0+a_1(x-x_0)+a_2(y-y_0)(x-x_1)+a_3(x-x_0)(y-y_1)(x-x_2)\\
&\quad +a_4(x,y)(y-y_0)(x-x_1)(y-y_2)(x-x_3).
\end{aligned}
$$

Thus, by considering the interpolation over Ω_3, we get from the above

$$
\begin{aligned}
a_3 &= \frac{f(x_3,y_3)-a_0-a_1(x_3-x_0)-a_2(y_3-y_0)(x_3-x_1)}{(x_3-x_0)(y_3-y_1)(x_3-x_2)}\\
&= \frac{\dfrac{f_{3,3}-f_{0,0}}{x_3-x_0}-[x_0,x_1;\ y_0,y_1]}{y_3-y_1}-[x_0,x_1,x_2;\ y_0,y_1,y_2]\frac{(y_3-y_0)(x_3-x_1)}{(x_3-x_0)(y_3-y_1)}\\
&= \left\{\frac{[x_0,x_1,x_3;\ y_0,y_1,y_3]_y-[x_0,x_1;\ y_0,y_1]}{y_3-y_1}-[x_0,x_1,x_2;\ y_0,y_1,y_2]\right.\\
&\quad \left.\cdot\frac{(y_3-y_0)(x_3-x_1)}{(x_3-x_0)(y_3-y_1)}\right\}\Big/(x_3-x_2)\\
&= [x_0,x_1,x_2,x_3;\ y_0,y_1,y_2,x_3].\\
a_4(x,y) &= [f(x,y)-a_0-a_1(x-x_0)-a_2(y-y_0)(x-x_1)\\
&\quad -a_3(x-x_0)(y-y_1)(x-x_2)]/[(y-y_0)(x-x_1)(y-y_2)(x-x_3)]\\
&= \left\{\frac{[x_0,x_1,x;\ y_0,y_1,y]-[x_0,x_1,x_2;\ y_0,y_1,y_2]}{y-y_2}-[x_0,\cdots,x_3;\ y_0,\cdots,y_3]\right.\\
&\quad \left.\cdot\frac{(x-x_0)(y-y_1)(x-x_2)}{(y-y_0)(x-x_1)(y-y_2)}\right\}\Big/(x-x_3)\\
&= \left\{[x_0,x_1,x_2,x;\ y_0,y_1,y_2,y]_y-[x_0,\cdots,x_3;\ y_0,\cdots,y_3]\right.\\
&\quad \left.\cdot\frac{(x-x_0)(y-y_1)(x-x_2)}{(y-y_0)(x-x_1)(y-y_2)}\right\}\Big/(x-x_3)\\
&= [x_0,\cdots,x_3,x;\ y_0,\cdots,y_3,y].
\end{aligned}
$$

Hence, the conclusion holds for $n=0,1,2,3$ in Theorems 2.1 and 2.2.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

Assume the results hold for the case of $\Omega_i,\ i = 0, 1, \cdots, 2n-1$. To be specific, the corresponding interpolating coefficients $a_i^{'}$s are calculated by means of Algorithm 2.1 and 2.2. And each $a_i(x, y)$ is defined in Theorem 2.1 if the i is odd. Otherwise, each $a_i(x, y)$ is defined in Theorem 2.2. Then on one hand, for $i = 2n$, it follows

$$
\begin{aligned}
a_{2n}(x,y) &= \{f(x,y) - a_0 - a_1(x-x_0) - a_2(y-y_0)(x-x_1) - \cdots \\
&\quad - a_{2n-1}(x-x_0)(y-y_1)\cdots(x-x_{2n-2})\}/[(y-y_0)\cdots(x-x_{2n-1})] \\
&= \cdots \\
&= \left\{ \frac{[x_0,\cdots,x_{2n-3},x;\ y_0,\cdots,y_{2n-3},y] - [x_0,\cdots,x_{2n-2};\ y_0,\cdots,y_{2n-2}]}{y_{2n}-y_{2n-2}} \right. \\
&\quad \left. - [x_0,\cdots,x_{2n-1};\ y_0,\cdots,y_{2n-1}]\frac{(x-x_0)(y-y_1)\cdots(x-x_{2n-2})}{(y-y_0)(x-x_1)\cdots(y-y_{2n-2})} \right\} \Big/ (x-x_{2n-1}) \\
&= \left\{ [x_0,\cdots,x_{2n-2},x;\ y_0,\cdots,y_{2n-2},y]_y - [x_0,\cdots,x_{2n-1};\ y_0,\cdots,y_{2n-1}] \right. \\
&\quad \left. \cdot \frac{(x-x_0)(y-y_1)\cdots(x-x_{2n-2})}{(y-y_0)(x-x_1)\cdots(y-y_{2n-2})} \right\} \Big/ (x-x_{2n-1}) \\
&= [x_0,\cdots,x_{2n-1},x;\ y_0,\cdots,y_{2n-1},y].
\end{aligned}
$$

On the other hand, simple computation show for $i = 2n+1$ that

$$
\begin{aligned}
a_{2n+1}(x,y) &= \{f(x,y) - a_0 - a_1(x-x_0) - a_2(y-y_0)(x-x_1) - \cdots - a_{2n}(y-y_0) \\
&\quad \cdots(x-x_{2n-1})\}/[(x-x_0)(y-y_1)\cdots(x-x_{2n-1})(y-y_{2n})] \\
&= \cdots \\
&= \left\{ \frac{[x_0,\cdots,x_{2n-2},x;\ y_0,\cdots,y_{2n-2},y] - [x_0,\cdots,x_{2n-1};\ y_0,\cdots,y_{2n-1}]}{y-y_{2n-1}} \right. \\
&\quad \left. - [x_0,\cdots,x_{2n};\ y_0,\cdots,y_{2n}]\frac{(y-y_0)(x-x_1)\cdots(y-y_{2n-1})}{(x-x_0)(y-y_1)\cdots(x-x_{2n-1})} \right\} \Big/ (x-x_{2n}) \\
&= \left\{ [x_0,\cdots,x_{2n-1},x;\ y_0,\cdots,y_{2n-1},y]_y - [x_0,\cdots,x_{2n};\ y_0,\cdots,y_{2n}] \right. \\
&\quad \left. \cdot \frac{(y-y_0)(x-x_1)\cdots(y-y_{2n-1})}{(x-x_0)(y-y_1)\cdots(x-x_{2n-1})} \right\} \Big/ (x-x_{2n}) \\
&= [x_0,\cdots,x_{2n},x;\ y_0,\cdots,y_{2n},y].
\end{aligned}
$$

As a result, (2.10) up to (2.15) is true by induction. □

When we turn to the corresponding interpolation problems in connection with bivariate polynomials $p_{2n}(x, y)$ and $p_{2n+1}(x, y)$ in (2.1) and (2.2), respectively, we find that many of the difficulties disappear due to the identity in Theorem 2.1 and 2.2. Hence, it is evident that we draw some conclusion as following.

Theorem 2.3. *For $i = 0, 1, \cdots, 2n$, the bivariate interpolating polynomial $p_{2n}(x, y)$ defined in* (2.1) *satisfies*

$$p_{2n}(x_i, y_i) = z_i = f(x_i, y_i) = f_{i,i}, \tag{2.16}$$

where the interpolating coefficients are defined in Algorithm 2.1*, i.e.,*

$$a_i = [x_0, \cdots, x_i;\ y_0, \cdots, y_i], \qquad i = 0, 1, \cdots, 2n.$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

Theorem 2.4. *The bivariate polynomial* $p_{2n+1}(x,y)$ *in the form of* (2.2) *interpolate over* Ω_{2n+1}, *i.e.,*

$$p_{2n+1}(x_i,y_i)=z_i=f(x_i,y_i)=f_{i,i},\qquad i=0,1,\cdots,2n+1, \tag{2.17}$$

where the corresponding interpolating coefficients are defined in Algorithm 2.1, *i.e.,*

$$a_i=[x_0,\cdots,x_i;\ y_0,\cdots,y_i],\qquad i=0,1,\cdots,2n+1.$$

Hence, we have computed the corresponding interpolating coefficients by making use of the recursion scheme in Algorithm 2.1 and 2.2, and have established the fundamental theorem with the bivariate polynomial interpolation over nonrectangular mesh. For the sake of illustration and convenience, we have computed the former explicit representation of some coefficients in (2.3) up to (2.7) in advance.

Now one may be interested in the dimension and bases of the interpolating polynomial spaces. Firstly, we shall denote by $\mathbb{P}_{n,n}(x,y)$ and $\mathbb{P}_{n+1,n}(x,y)$ the bivariate polynomial spaces in the form of product tensor, respectively.

$$\mathbb{P}_{n,n}=\operatorname{span}\{1,x,\cdots,x^n\}\bigotimes\{1,y,\cdots,y^n\},$$
$$\mathbb{P}_{n+1,n}=\operatorname{span}\{1,x,\cdots,x^{n+1}\}\bigotimes\{1,y,\cdots,y^n\}.$$

Secondly, one can count the number of terms in (2.1) and (2.2), which equals to the number of the interpolating conditions over Ω_{2n} and Ω_{2n+1}, respectively. Meanwhile, we can check that all the conditions are independent of each other. Also it is easy to check that the bivariate polynomials in all terms of (2.1) and (2.2) are linear independent, respectively, due to their degrees. Hence, it follows that the interpolating polynomials in the form of (2.1) and (2.2) are determined uniquely over arbitrary triple Ω_{2n} and Ω_{2n+1}, respectively.

Theorem 2.5. *The interpolating polynomial* $p_{2n}(x,y)$ *is determined over* Ω_{2n} *uniquely in the form of* (2.1), *and the dimension of the interpolating polynomial space is* $2n+1$, *i.e.,*

$$\begin{aligned}p_{2n}(x,y)\ \in \operatorname{span}\Big\{&1,x-x_0,(y-y_0)(x-x_1),(x-x_0)(y-y_1)(x-x_2),\cdots,\\&(y-y_0)(x-x_1)\cdots(y-y_{2n-2})(x-x_{2n-1})\Big\}\subset\mathbb{P}_{n,n}.\end{aligned} \tag{2.18}$$

Theorem 2.6. *The interpolating polynomial* $p_{2n+1}(x,y)$ *with the representation* (2.2) *is determined uniquely over* Ω_{2n+1}, *i.e., the corresponding interpolating coefficients are computed uniquely with Algorithm* 2.1 *and* 2.2. *Also the dimension of the interpolating polynomial space is* $2n+2$, *i.e.,*

$$\begin{aligned}p_{2n+1}(x,y)\in \operatorname{span}\Big\{&1,x-x_0,(y-y_0)(x-x_1),(x-x_0)(y-y_1)(x-x_2),\cdots,\\&(y-y_0)(x-x_1)\cdots(y-y_{2n-2})(x-x_{2n-1}),(x-x_0)(y-y_1)\\&\cdots(x-x_{2n})\Big\}\subset\mathbb{P}_{n+1,n}.\end{aligned} \tag{2.19}$$

Remark 2.2. The representation of $p_{2n}(x,y)$ and $p_{2n+1}(x,y)$ is related to the order of the nodes.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

3. Error estimation of the bivariate polynomial interpolation over Ω_{2n}

In this section, we shall use Theorem 2.1 to establish the corresponding error estimation of the bivariate interpolation over Ω_{2n}, and the relation between high-order partial derivatives and the non-tensor-product-type divided differences with even order.

Suppose that there exists a one-to-one projection such that

$$t \leftrightarrow (x(t), y(t)), \quad t \in [\alpha, \beta], \quad t_i \leftrightarrow (x(t_i), y(t_i)) \equiv (x_i, y_i), \quad i = 0, 1, \cdots, 2n.$$

Theorem 3.1. *Suppose that $f(x, y)$ is a $2n+1$-differentiable function with two variables in the domain $D \supset \Omega_{2n}$, and that $x(t), y(t)$ are $2n+1$-differentiable functions with variable t. Then $\forall (x, y) \in C : \ x = x(t), y = y(t)$, and $\Omega_{2n} \subset C \subset D$, there exists some number $\tau \in I(t, t_0, t_1, \cdots, t_{2n})$, such that*

$$E[f] = f(x, y) - p_{2n}(x, y) = K(x, y) \prod_{i=0}^{n} (x - x_{2i}) \prod_{j=1}^{n} (y - y_{2j-1}), \tag{3.1}$$

where $I(t, t_0, t_1, \cdots, t_{2n})$ is denoted by the minimal open interval containing $t, t_0, t_1, \cdots, t_{2n}$, and

$$K(x, y) = \frac{\dfrac{\mathrm{d}^{2n+1}}{\mathrm{d}s^{2n+1}} \Big(f(x(s), y(s)) - p_{2n}(x(s), y(s)) \Big)}{\dfrac{\mathrm{d}^{2n+1}}{\mathrm{d}s^{2n+1}} \Big(\prod_{i=0}^{n} (x(s) - x_{2i}) \prod_{j=1}^{n} (y(s) - y_{2j-1}) \Big)} \Bigg|_{s=\tau}, \tag{3.2}$$

Proof. For the sake of convenience, we denote by

$$u(t) = f(x(t), y(t)), \quad K(x, y) = [x_0, \cdots, x_{2n}, x;\ y_0, \cdots, y_{2n}, y].$$

In order to simplify the bivariate non-tensor-product divided difference $K(x, y)$, we construct a function with variable s

$$\Phi(s) = u(s) - p_{2n}(x(s), y(s)) - K(x, y) \prod_{i=0}^{n} \Big(x(s) - x_{2i} \Big) \prod_{j=1}^{n} \Big(y(s) - y_{2j-1} \Big). \tag{3.3}$$

And there exist $2n + 2$ zeros $s = t, t_0, t_1, \cdots, t_{2n}$ for the function $\Phi(s)$. Hence, by using Rolle Theorem for $2n + 1$ times, it is evident that

$$\Phi^{(2n+1)}(\tau) = 0, \qquad \tau \in I(t, t_0, t_1, \cdots, t_{2n}),$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

i.e.,

$$\begin{aligned}&\Bigg[u^{2n+1}(s)-\frac{\mathrm{d}^{2n+1}}{\mathrm{d}s^{2n+1}}p_{2n}(x(s),y(s))\\&\qquad-K(x,y)\frac{\mathrm{d}^{2n+1}}{\mathrm{d}s^{2n+1}}\Bigg(\prod_{i=0}^{n}(x(s)-x_{2i})\prod_{j=1}^{n}(y(s)-y_{2j-1})\Bigg)\Bigg]_{s=\tau}=0\\\Rightarrow\quad K(x,y)=&\frac{\dfrac{\mathrm{d}^{2n+1}}{\mathrm{d}s^{2n+1}}\Big(f(x(s),y(s))-p_{2n}(x(s),y(s))\Big)}{\dfrac{\mathrm{d}^{2n+1}}{\mathrm{d}s^{2n+1}}\Bigg(\displaystyle\prod_{i=0}^{n}(x(s)-x_{2i})\prod_{j=1}^{n}(y(s)-y_{2j-1})\Bigg)}\Bigg|_{s=\tau}.\end{aligned}$$

As a result, by means of the identity (2.1), it follows that the error estimate $\forall(x,y)\in C:\ x=x(t),y=y(t),\ C\supset\Omega_{2n}$,

$$E[f]=f(x,y)-p_{2n}(x,y)=K(x,y)\prod_{i=0}^{n}(x-x_{2i})\prod_{j=1}^{n}(y-y_{2j-1}),$$

where $K(x,y)$ is defined in (3.2). This proves the theorem. □

In order to calculate $K(x,y)$ in (3.2), we shall consider a special case and work out much specifical error estimation provided that $\Omega_{2n}\subset C:\ x(t)=t,y(t)=kt+b$ with $k\neq 0$.

Theorem 3.2. *Suppose that $f(x,y)$ is a $2n+1$-differentiable function with two variables in the domain $D\supset\Omega_{2n}$. Then $\forall(x,y)\in C:\ x(t)=t,y(t)=kt+b,\ k\neq 0$, and $\Omega_{2n}\subset C$, there exists some number $\tau\in I(t,t_0,t_1,\cdots,t_{2n})$, such that*

$$\begin{aligned}E[f]=&f(x,y)-p_{2n}(x,y)\\=&\frac{\left(\dfrac{\partial}{\partial x}+k\dfrac{\partial}{\partial y}\right)^{2n+1}f(\xi,\eta)}{(2n+1)!k^n}\prod_{i=0}^{n}(x-x_{2i})\prod_{j=1}^{n}(y-y_{2j-1}),\end{aligned}\tag{3.4}$$

where $\xi=x(\tau),\ \eta=y(\tau),\ (\xi,\eta)\in C\subset D$.

As a matter of fact, the proof proceeds along the same lines as the proofs of Theorem 3.1. Firstly, by directly computing, one obtains the interpolating polynomial with the presentation

$$p_{2n}(x(s),y(s))=a_{2n}(s^n+\cdots)(k^ns^n+\cdots)+\text{polynomial}\ \ \text{degree}<2n,$$

which implies

$$\frac{\mathrm{d}^{2n+1}}{\mathrm{d}s^{2n+1}}p_{2n}(x(s),y(s))=0.\tag{3.5}$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

Similarly, we have

$$\frac{\mathrm{d}^{2n+1}}{\mathrm{d}s^{2n+1}}\left[\prod_{i=0}^{n}(s-x_{2i})\prod_{j=1}^{n}(ks+b-y_{2j-1})\right]=(2n+1)!k^n. \tag{3.6}$$

Secondly, we perform for the order $2n+1$ of the derivative by introduction and get

$$\frac{\mathrm{d}^{2n+1}}{\mathrm{d}s^{2n+1}}f(s,ks+b)=\left(\frac{\partial}{\partial x}+k\frac{\partial}{\partial y}\right)^{2n+1}f(x,y), \tag{3.7}$$

Thus by means of Theorem 3.1, (3.5) up to (3.7) implies

$$K(x,y)=\frac{\left(\dfrac{\partial}{\partial x}+k\dfrac{\partial}{\partial y}\right)^{2n+1}f(\xi,\eta)}{(2n+1)!k^n}, \tag{3.8}$$

where $\xi=x(\tau),\ \eta=y(\tau),\ (\xi,\eta)\in C$. Therefore, this shows that it is valid for Theorem 3.2.

Analogous to the proof of Theorem 3.2, it is evident that

Corollary 3.1 *Suppose that $f(x,y)$ is a $2n+1$-differentiable function with two variables in the domain $D\supset\Omega_{2n}$. Then $\forall(x,y)\in C:\ x(t)=\lambda t+b_0, y(t)=\mu t+b_1,\ \lambda\mu\neq 0$, and $\Omega_{2n}\subset C$, there exists some number $\tau\in I(t,t_0,t_1,\cdots,t_{2n})$, such that*

$$\begin{aligned}E[f]=&f(x,y)-p_{2n}(x,y)\\=&\frac{\left(\lambda\dfrac{\partial}{\partial x}+\mu\dfrac{\partial}{\partial y}\right)^{2n+1}f(\xi,\eta)}{(2n+1)!\lambda^{n+1}\mu^n}\prod_{i=0}^{n}(x-x_{2i})\prod_{j=1}^{n}(y-y_{2j-1}),\end{aligned} \tag{3.9}$$

where $\xi=x(\tau),\ \ \eta=y(\tau),\ \ (\xi,\eta)\in C\subset D$.

It may be noted that the relation shall be established between derivatives with high order and bivariate divided differences in Algorithm 2.1 and 2.2.

Theorem 3.3. *Suppose that $f(x,y)$ is a $2n$-differentiable function with two variables in the domain $D\supset\Omega_{2n}$. Then $\forall(x,y)\in C:\ x(t)=t, y(t)=kt+b,\ k\neq 0$, and $\Omega_{2n}\subset C$, there exists some number $\tau\in I(t,t_0,t_1,\cdots,t_{2n})$, such that*

$$[x_0,\cdots,x_{2n};\ y_0,\cdots,y_{2n}]=\frac{\left(\dfrac{\partial}{\partial x}+k\dfrac{\partial}{\partial y}\right)^{2n}f(\xi,\eta)}{(2n)!k^n}, \tag{3.10}$$

where $\xi=x(\tau),\ \ \eta=y(\tau),\ \ (\xi,\eta)\in C\subset D$.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

Proof. For the sake of convenience, we construct a function

$$F(x,y) = f(x,y) - p_{2n}(x,y), \quad (x,y) \in C \subset D.$$

And there exist $2n+1$ zeros $s = t_0, t_1, \cdots, t_{2n}$ for the function $F(x(t), y(t))$. Hence, on one hand, by means of Rolle Theorem for $2n$ times, it follows that there exists some $\tau \in I(t_0, t_1, \cdots, t_{2n})$ such that

$$\begin{aligned} & \frac{\mathrm{d}^{2n} F(x(t), y(t))}{\mathrm{d}t^{2n}}\bigg|_{t=\tau} = 0, \\ \Rightarrow \quad & \frac{\mathrm{d}^{2n} f(x(t), y(t))}{\mathrm{d}t^{2n}}\bigg|_{t=\tau} = \frac{\mathrm{d}^{2n} p_{2n}(x(t), y(t))}{\mathrm{d}t^{2n}}\bigg|_{t=\tau}. \end{aligned}$$

On the other hand, by directly computing, we get

$$\begin{aligned} & \frac{\mathrm{d}^{2n}}{\mathrm{d}t^{2n}} f(t, kt+b) = \left(\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right)^{2n} f(x,y), \\ & \frac{\mathrm{d}^{2n}}{\mathrm{d}t^{2n}} p_{2n}(x(t), y(t)) = [x_0, \cdots, x_{2n};\ y_0, \cdots, y_{2n}] \cdot k^n (2n)!, \end{aligned}$$

where

$$\begin{aligned} p_{2n}(x(t), y(t)) = {} & [x_0, \cdots, x_{2n};\ y_0, \cdots, y_{2n}](t^n + \cdots)(k^n t^n + \cdots) \\ & + \text{polynomial degree} < 2n. \end{aligned}$$

Finally, the property (3.10) follows directly from the above. □

Analogous to the proof of Theorem 3.2, it is evident that

Corollary 3.2 *Suppose that $f(x,y)$ is a $2n$-differentiable function with two variables in the domain $D \supset \Omega_{2n}$. Then $\forall (x,y) \in C:\ x(t) = \lambda t + b_0, y(t) = \mu t + b_1,\ \lambda\mu \neq 0$, and $\Omega_{2n} \subset C$, there exists some number $\tau \in I(t_0, t_1, \cdots, t_{2n})$, such that*

$$[x_0, \cdots, x_{2n};\ y_0, \cdots, y_{2n}] = \frac{\left(\lambda\dfrac{\partial}{\partial x} + \mu\dfrac{\partial}{\partial y}\right)^{2n} f(\xi, \eta)}{(2n)!\lambda^n \mu^n}, \tag{3.11}$$

where $\xi = x(\tau),\ \eta = y(\tau),\ (\xi, \eta) \in C \subset D$.

4. Error estimate for bivariate polynomial interpolation over Ω_{2n+1}

In this section, we shall use Theorem 2.2 to establish the corresponding error estimation of the bivariate polynomial interpolation over Ω_{2n+1}, and obtain an equivalence between high-order derivatives and bivariate divided differences with odd order. All the analysis is similar but different with the case of Ω_{2n}.

Suppose that there exists a one-to-one projection such that

$$t \leftrightarrow (x(t), y(t)),\ \ t \in [\alpha, \beta],\ \ t_i \leftrightarrow (x(t_i), y(t_i)) \equiv (x_i, y_i),\ \ i = 0, 1, \cdots, 2n+1.$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

Theorem 4.1. *Suppose that $f(x,y)$ is a $2n+2$-differentiable function with two variables in the domain $D \supset \Omega_{2n+1}$, and that $x(t), y(t)$ are $2n+2$-differentiable functions with variable t. Then $\forall (x,y) \in C:\ x = x(t), y = y(t)$, and $\Omega_{2n+1} \subset C \subset D$, there exists some number $\tau \in I(t, t_0, t_1, \cdots, t_{2n+1})$, such that*

$$E[f] = f(x,y) - p_{2n+1}(x,y) = M(x,y) \prod_{i=0}^{n} (x - x_{2i+1})(y - y_{2i}), \tag{4.1}$$

where $p_{2n+1}(x,y)$ is defined as (2.2)*, $I(t, t_0, t_1, \cdots, t_{2n+1})$ is denoted by the minimal open interval containing $t, t_0, t_1, \cdots, t_{2n+1}$, and*

$$M(x,y) = \left. \frac{\dfrac{\mathrm{d}^{2n+2}}{\mathrm{d}s^{2n+2}} \Big(f(x(s), y(s)) - p_{2n+1}(x(s), y(s)) \Big)}{\dfrac{\mathrm{d}^{2n+2}}{\mathrm{d}s^{2n+2}} \Big(\displaystyle\prod_{i=0}^{n} (x(s) - x_{2i+1})(y(s) - y_{2i}) \Big)} \right|_{s=\tau}, \tag{4.2}$$

Proof. For the sake of convenience, we denote by

$$u(t) = f(x(t), y(t)), \quad M(x,y) = [x_0, \cdots, x_{2n+1}, x;\ y_0, \cdots, y_{2n+1}, y].$$

In order to simplify the bivariate non-tensor-product divided difference $M(x,y)$, we construct a function with variable s

$$\Psi(s) = u(s) - p_{2n+1}(x(s), y(s)) - M(x,y) \prod_{i=0}^{n} (x(s) - x_{2i+1})(y(s) - y_{2i}). \tag{4.3}$$

And there exist $2n+3$ zeros $s = t, t_0, t_1, \cdots, t_{2n+1}$ for the function $\Psi(s)$. Hence, with the help of Rolle Theorem for $2n+2$ times, it is evident that

$$\Psi^{(2n+2)}(\tau) = 0, \qquad \tau \in I(t, t_0, t_1, \cdots, t_{2n}),$$

i.e.,

$$\begin{aligned} &\Big[u^{2n+2}(s) - \frac{\mathrm{d}^{2n+2}}{\mathrm{d}s^{2n+2}} p_{2n+1}(x(s), y(s)) \\ &\qquad - M(x,y) \frac{\mathrm{d}^{2n+2}}{\mathrm{d}s^{2n+2}} \Big(\prod_{i=0}^{n} (x(s) - x_{2i+1})(y(s) - y_{2i}) \Big) \Big]_{s=\tau} = 0 \\ \Rightarrow \quad & M(x,y) = \left. \frac{\dfrac{\mathrm{d}^{2n+2}}{\mathrm{d}s^{2n+2}} \Big(f(x(s), y(s)) - p_{2n+1}(x(s), y(s)) \Big)}{\dfrac{\mathrm{d}^{2n+2}}{\mathrm{d}s^{2n+2}} \Big(\displaystyle\prod_{i=0}^{n} (x(s) - x_{2i+1})(y(s) - y_{2i}) \Big)} \right|_{s=\tau}. \end{aligned}$$

As a result, by means of the identity (2.2), it follows that the error estimate $\forall (x,y) \in C:\ x = x(t), y = y(t),\ C \supset \Omega_{2n+1}$,

$$E[f] = f(x,y) - p_{2n+1}(x,y) = M(x,y) \prod_{i=0}^{n} (x - x_{2i+1})(y - y_{2i}),$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

where $M(x,y)$ is defined in (4.2). This proves the theorem. □

We shall work out much specifical error estimation provided that $\Omega_{2n+1} \subset C:\ x(t) = t, y(t) = kt + b$ with $k \neq 0$. Thus we can calculate $M(x,y)$ in (4.2) in a specific way.

Theorem 4.2. *Suppose that $f(x,y)$ is a $2n+2$-differentiable function with two variables in the domain $D \supset \Omega_{2n+1}$. Then $\forall (x,y) \in C:\ x(t) = t, y(t) = kt + b,\ k \neq 0$, and $\Omega_{2n+1} \subset C$, there exists some number $\tau \in I(t, t_0, t_1, \cdots, t_{2n+1})$, such that*

$$\begin{aligned} E[f] =& f(x,y) - p_{2n+1}(x,y) \\ =& \frac{\left(\dfrac{\partial}{\partial x} + k\dfrac{\partial}{\partial y}\right)^{2n+2} f(\xi,\eta)}{(2n+2)!k^n} \prod_{i=0}^{n}(x - x_{2i+1})(y - y_{2i}), \end{aligned} \tag{4.4}$$

where $\xi = x(\tau),\ \ \eta = y(\tau),\ \ (\xi,\eta) \in C \subset D$.

Proof. The proof is analogous to the proof of Theorem 4.1. On one hand, by means of direct computation, it follows that the representation of the interpolating polynomial

$$p_{2n+1}(x(s), y(s)) = a_{2n}(s^{n+1} + \cdots)(k^n s^n + \cdots) + \text{polynomial}\ \ \text{degree} < 2n+1,$$

implies

$$\frac{\mathrm{d}^{2n+2}}{\mathrm{d}s^{2n+2}} p_{2n+2}(x(s), y(s)) = 0. \tag{4.5}$$

Similarly, we get

$$\frac{\mathrm{d}^{2n+2}}{\mathrm{d}s^{2n+2}}\left[\prod_{i=0}^{n}(s - x_{2i+1})(ks + b - y_{2i})\right] = (2n+2)!k^{n+1}. \tag{4.6}$$

On the other hand, by performing for the order $2n+2$ of the derivative by introduction, we obtain

$$\frac{\mathrm{d}^{2n+2}}{\mathrm{d}s^{2n+2}} f(s, ks + b) = \left(\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right)^{2n+2} f(x,y), \tag{4.7}$$

Thus with the help of Theorem 4.1, (4.5) up to (4.7) implies

$$M(x,y) = \frac{\left(\dfrac{\partial}{\partial x} + k\dfrac{\partial}{\partial y}\right)^{2n+2} f(\xi,\eta)}{(2n+2)!k^{n+1}}, \tag{4.8}$$

where $\xi = x(\tau),\ \ \eta = y(\tau),\ \ (\xi,\eta) \in C \subset D$. Therefore, this proves Theorem 4.2. □

Analogous to the proof of Theorem 4.2, it is easily followed that

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

Corollary 4.1 *Provided that $f(x,y)$ is a $2n+2$-differentiable function with two variables in the domain $D \supset \Omega_{2n+1}$, then $\forall (x,y) \in C:\ x(t) = \lambda t + b_0, y(t) = \mu t + b_1,\ \lambda\mu \neq 0$, and $\Omega_{2n} \subset C$, there exists some number $\tau \in I(t, t_0, t_1, \cdots, t_{2n})$, such that*

$$\begin{aligned} E[f] =& f(x,y) - p_{2n+1}(x,y) \\ =& \frac{\left(\lambda\dfrac{\partial}{\partial x} + \mu\dfrac{\partial}{\partial y}\right)^{2n+2} f(\xi,\eta)}{(2n+2)!\lambda^{n+1}\mu^{n+1}} \prod_{i=0}^{n}(x - x_{2i+1})(y - y_{2i}), \end{aligned} \tag{4.9}$$

where

$$\xi = x(\tau),\quad \eta = y(\tau),\quad (\xi,\eta) \in C \subset D.$$

The procedure of the proofs for Theorem 4.1 and 4.2 is useful for constructing the relation between high-order derivatives and bivariate high-odd-order divided differences in Algorithm 2.1 and 2.2.

Theorem 4.3. *Suppose that $f(x,y)$ is a $2n+1$-differentiable function with two variables in the domain $D \supset \Omega_{2n+1}$. Then $\forall (x,y) \in C:\ x(t) = t, y(t) = kt + b,\ k \neq 0$, and $\Omega_{2n+1} \subset C$, there exists some number $\tau \in I(t, t_0, t_1, \cdots, t_{2n+1})$, such that*

$$[x_0, \cdots, x_{2n+1};\ y_0, \cdots, y_{2n+1}] = \frac{\left(\dfrac{\partial}{\partial x} + k\dfrac{\partial}{\partial y}\right)^{2n+1} f(\xi,\eta)}{(2n+1)!k^{n+1}}, \tag{4.10}$$

where $\xi = x(\tau),\ \ \eta = y(\tau),\ \ (\xi,\eta) \in C \subset D$.

Proof. Firstly, we begin with a function

$$F(x,y) = f(x,y) - p_{2n}(x,y), \quad (x,y) \in C \subset D,$$

which has $2n+2$ zeros $s = t_0, t_1, \cdots, t_{2n+1}$. Hence, based on Rolle Theorem for $2n+1$ times, it is evident that there exists some $\tau \in I(t_0, t_1, \cdots, t_{2n+1})$ such that

$$\begin{aligned} & \frac{\mathrm{d}^{2n+1}F(x(t),y(t))}{\mathrm{d}t^{2n+1}}\bigg|_{t=\tau} = 0, \\ \Rightarrow \quad & \frac{\mathrm{d}^{2n+1}f(x(t),y(t))}{\mathrm{d}t^{2n+1}}\bigg|_{t=\tau} = \frac{\mathrm{d}^{2n+1}p_{2n+1}(x(t),y(t))}{\mathrm{d}t^{2n+1}}\bigg|_{t=\tau}. \end{aligned}$$

Secondly, some conclusion follows if we differentiate the functions $f(t, kt+b)$ and $p_{2n+1}(x(t), y(t))$ $2n+1$ times with respect to t.

$$\begin{aligned} & \frac{\mathrm{d}^{2n+1}}{\mathrm{d}t^{2n+1}} f(t, kt+b) = \left(\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right)^{2n+1} f(x,y), \\ & \frac{\mathrm{d}^{2n+1}}{\mathrm{d}t^{2n}} p_{2n+1}(x(t),y(t)) = [x_0, \cdots, x_{2n+1};\ y_0, \cdots, y_{2n+1}] \cdot k^{n+1}(2n+1)!, \end{aligned}$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

where

$$p_{2n+1}(x(t),y(t)) = [x_0,\cdots,x_{2n+1};\ y_0,\cdots,y_{2n+1}](t^{n+1}+\cdots)(k^n t^n+\cdots) + \text{polynomial degree} < 2n+1.$$

Finally, one can easily prove the validity of (4.10). □

With a similar proof of Theorem 4.2, it is clear that

Corollary 4.2 *Suppose that $f(x,y)$ is a $2n+1$-differentiable function with two variables in the domain $D \supset \Omega_{2n+1}$. Then $\forall (x,y) \in C:\ x(t) = \lambda t + b_0, y(t) = \mu t + b_1,\ \lambda\mu \neq 0$, and $\Omega_{2n+1} \subset C$, there exists some number $\tau \in I(t_0,t_1,\cdots,t_{2n+1})$, such that*

$$[x_0,\cdots,x_{2n+1};\ y_0,\cdots,y_{2n+1}] = \frac{\left(\lambda\dfrac{\partial}{\partial x} + \mu\dfrac{\partial}{\partial y}\right)^{2n+1} f(\xi,\eta)}{(2n+1)!\lambda^{n+1}\mu^n}, \tag{4.11}$$

where $\xi = x(\tau),\ \ \eta = y(\tau),\ \ (\xi,\eta) \in C \subset D$.

5. Numerical examples

In this section, by using Algorithm 2.1 and 2.2, we determine some low-order polynomial interpolation over nonrectangular meshes. Firstly, we compute the representation of these low-order polynomials. Secondly, these figures of the polynomials with different degrees are plotted with Matlab 6.5.

We shall make a preparation with some denotation for the following numerical examples. We denote by $\Omega_k = \{(x_0,y_0),(x_1,y_1),\cdots,(x_k,y_k)\}$, where $k = 4,5$. Then by (2.1) and (2.2), we shall give the explicit representation of the interpolating polynomials

$$\begin{aligned} p_2(x,y) &= a_0 + a_1(x-x_0) + a_2(y-y_0)(x-x_1),\\ p_3(x,y) &= p_2(x,y) + a_3(x-x_0)(y-y_1)(x-x_2),\\ p_4(x,y) &= p_3(x,y) + a_4(y-y_0)(x-x_1)(y-y_2)(x-x_3),\\ p_5(x,y) &= p_4(x,y) + a_5(x-x_0)(y-y_1)(x-x_2)(y-y_3)(x-x_4). \end{aligned} \tag{5.1}$$

Example 5.1. In one case I, suppose $P_0(-7.5,-9.5)$, $P_1(-5,-4)$, $P_2(-3,-2)$, $P_3(0.2,-1)$, $P_4(4,2)$, which are shown in Fig. 1. After we input the corresponding vertical coordinate z_i, $i = 0,\cdots,4$ in Table 2, we can work out the five interpolating coefficients as shown in Table 3 by using Algorithm 2.1. As a result, one can determine the interpolating polynomial $p_4(x,y)$ as shown in Fig. 2, where $(x,y) \in [-8,8] \times [-10,10]$.

In the other case II, we consider the interpolating nodes $P_0(-0.75,-0.95)$, $P_1(-0.5,-0.4)$, $P_2(-0.3,-0.2)$, $P_3(0.02,-0.1)$, $P_4(0.4,0.2)$ as shown in Fig. 3. For the corresponding vertical coordinate z_i, $i = 0,\cdots,4$ in Table 2, we calculate the five interpolating coefficients as shown in Table 3 by using Algorithm 2.1. Hence, we determine the interpolating polynomial $p_4(x,y)$ as shown in Fig. 4, where $(x,y) \in [-0.8,0.8] \times [-1,1]$.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

Table 2: The vertical coordinates $z_i,\ i = 0, \cdots, 4$ in Example 5.1.

Clusters	A (eV)	I (eV)	χ(eV)	η(eV)	ω(eV)
Vertical coordinate	z_0	z_1	z_2	z_3	z_4
One case I	0.0057	−0.0519	−0.1918	0.8355	−0.1036
The other case II	0.7858	0.9547	0.9848	0.9999	0.9733

Table 3: The coefficients of the interpolating polynomials $p_4(x, y)$'s in Example 5.1.

Clusters	A (eV)	I (eV)	χ(eV)	η(eV)	ω(eV)
Coefficients	a_0	a_1	a_2	a_3	a_4
Values in I	0.0057	−0.0230	−0.0063	0.0174	−0.0048
Values in II	0.7858	0.6756	−0.7001	0.0453	0.7200

Table 4: The vertical coordinates $z_i,\ i = 0, \cdots, 5$ in Example 5.2.

Vertical coordinate	z_0	z_1	z_2	z_3	z_4	z_5
One case A	1.4451	1.2202	1.4020	1.3475	1.1599	1.5059
The other case B	1.0259	1.0472	1.0845	1.0335	1.0244	1.0890

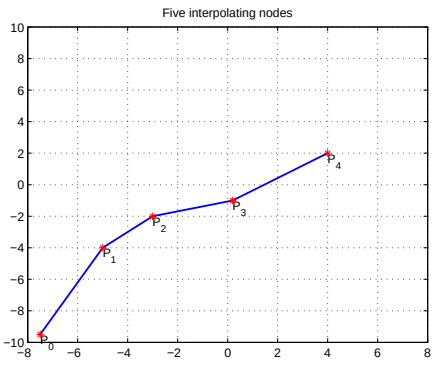


Figure 1: Ω_4 with Five interpolating nodes.

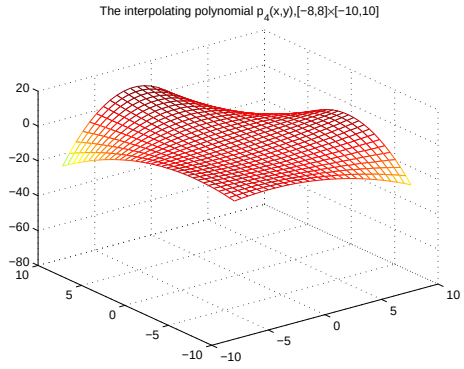


Figure 2: The interpolating polynomial $p_4(x, y)$ over Ω_4, $(x, y) \in [-8, 8] \times [-10, 10]$.

Table 5: The coefficients of the interpolating polynomials $p_4(x, y)$'s in Example 5.2.

Coefficients	a_0	a_1	a_2	a_3	a_4	a_5
Values in A	1.4451	−0.4498	1.2548	−1.7941	−0.4573	1.5552
Values in B	1.0259	0.0852	0.1302	2.8474	−18.6358	29.4940

Example 5.2. In one case A, suppose $P_0(-0.9, 0.2)$, $P_1(-0.4, 0.6)$, $P_2(0.2, 0.8)$, $P_3(0.8, 0.4)$, $P_4(0.6, -0.4)$, $P_5(-0.6, -0.8)$ as shown in Fig. 5. By considering the corresponding vertical coordinate z_i, $i = 0, \cdots, 5$ in Table 4, and using Algorithm 2.2, we can work out the six interpolating coefficients as shown in Table 5. Thus we obtain the interpolating polynomial $p_5(x, y)$ which is plotted in Fig. 6, where $(x, y) \in [-1, 1]^2$.

In the other case B, we consider the interpolating nodes $P_0(-0.45, 0.1)$, $P_1(-0.2, 0.3)$,

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

Table 6: The vertical coordinates $z_i,\ i = 0, \cdots, 5$ in Example 5.3.

Vertical coordinate	z_0	z_1	z_2	z_3	z_4	z_5
Case I	0.9802	0.9869	−0.0052	0.9856	0.9838	0.9896
Case II	−0.0052	0.9856	0.9802	0.9869	0.9838	0.9896

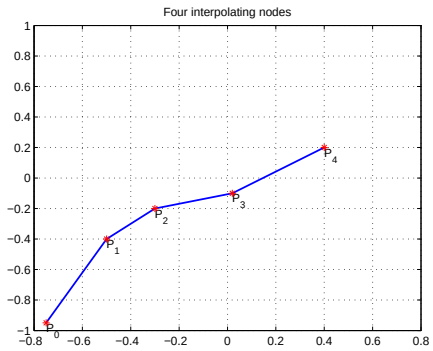


Figure 3: Another Ω_4 with Five interpolating nodes.

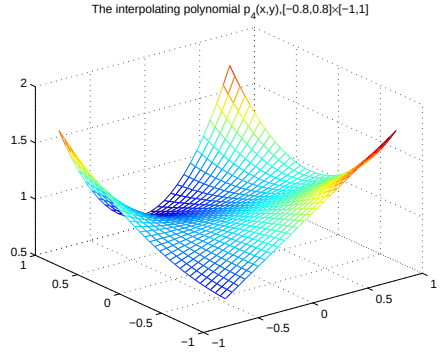


Figure 4: The interpolating polynomial $p_4(x, y)$ over $\Omega_4,\ (x, y) \in [-0.8, 0.8] \times [-1, 1]$.

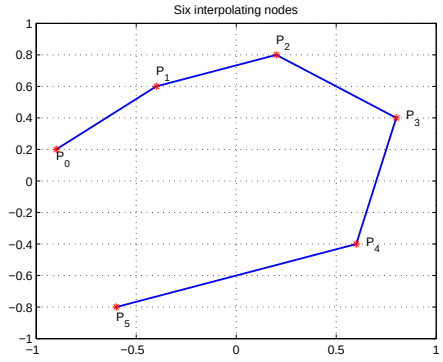


Figure 5: Ω_5 with six interpolating nodes.

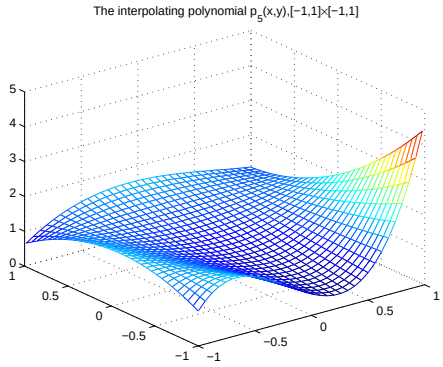


Figure 6: The interpolating polynomial $p_5(x, y)$ over $\Omega_5,\ (x, y) \in [-1, 1]^2$.

$P_2(0.1, 0.4),\ P_3(0.4, 0.2),\ P_4(0.3, -0.2),\ P_5(-0.3, -0.4)$ which are plotted in Fig. 7. For the corresponding vertical coordinate $z_i,\ i = 0, \cdots, 5$ in Table 4, by using Algorithm 2.2, we calculate the six interpolating coefficients in Table 5. And then we plot the interpolating polynomial $p_5(x, y)$ in Fig. 8, where $(x, y) \in [-0.5, 0.5]^2$.

Example 5.3. On one hand, let $P_0(-1.5, 0.8),\ P_1(-1, -1.6),\ P_2(0.1, 0.1),\ P_3(0.4,\ 1.8),\ P_4(0.8, -1.6),\ P_5(1.6, 1.2)$ in Case I, which are shown in Fig. 9. By means of the given vertical coordinates $z_i,\ i = 0, \cdots, 5$ in Table 6, and Algorithm 2.2, the six interpolating coefficients are worked out as in Table 7. Hence, the four interpolating polynomials are determined over Ω_5, which is plotted in Fig. 10.

On the other hand, we change the order of the interpolating nodes as $P_0(0.1, 0.1),$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

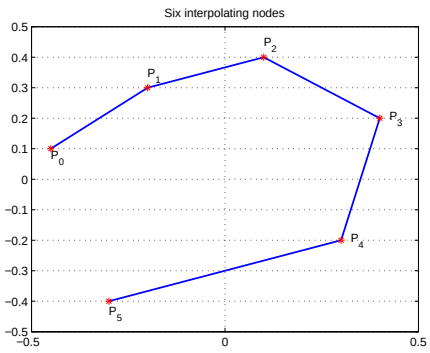


Figure 7: Another Ω_5 with six interpolating nodes.

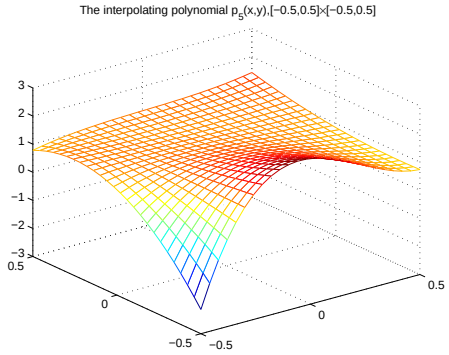


Figure 8: The interpolating polynomial $p_5(x, y)$ over Ω_5, $(x, y) \in [-0.5, 0.5]^2$.

Table 7: The coefficients of the interpolating polynomials $p_5(x, y)$ in case I and II in Example 5.3.

Coefficients	a_0	a_1	a_2	a_3	a_4	a_5
Case I	0.9802	0.0135	1.3078	−0.9551	1.9139	−1.3466
Case II	−0.0052	3.3030	−4.7144	8.4736	14.2483	−1.0152

$P_1(0.4, 1.8)$, $P_2(-1.5, 0.8)$, $P_3(-1, -1.6)$, $P_4(0.8, -1.6)$, $P_5(1.6, 1.2)$ in Fig. 11 in Case II, and consider the corresponding vertical coordinates z_i, $i = 0, \cdots, 5$ in Table 6. And then we calculate the interpolating coefficients in Table 7 and plot the figures of the interpolating polynomials as Fig. 12.

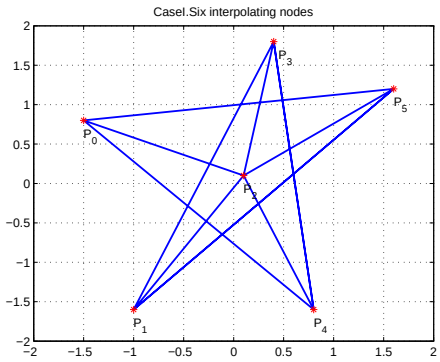


Figure 9: Ω_5 with Six interpolating nodes in Case I.

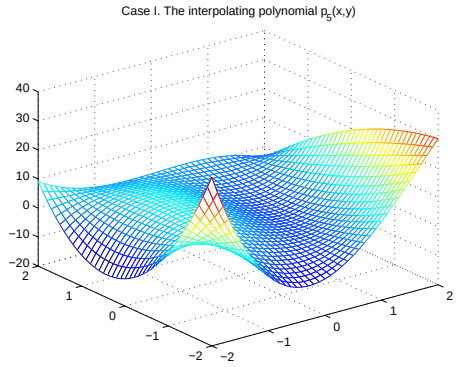


Figure 10: The interpolating polynomial $p_5(x, y)$ over Ω_5 in Case I.

Remark 5.1. From Case I and Case II in Example 5.3, it follows that the interpolating polynomial changes as the order of the interpolating nodes, although the interpolating node collection is invariant.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

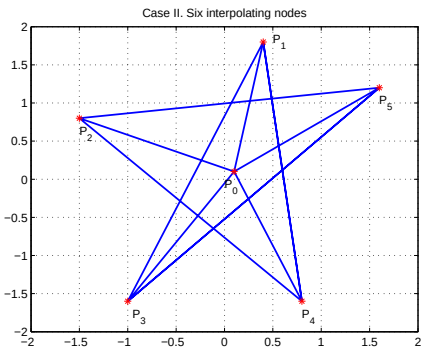


Figure 11: Ω_5 with Six interpolating nodes in Case II.

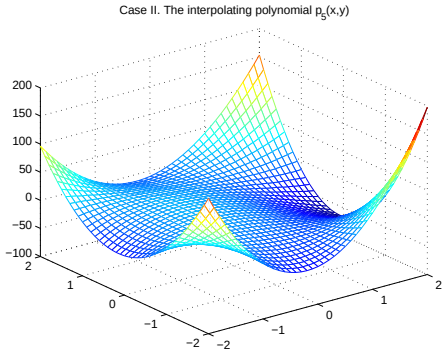


Figure 12: The interpolating polynomial $p_5(x, y)$ over Ω_5 in Case II.

6. Computational complexity of the bivariate polynomial interpolation

Now that we have learned the "how" of the bivariate non-tensor-product-typed continued fraction interpolation, here are a few words about "why". Since classic method for scattered data interpolation involves radial basis functions, we shall make analysis of the computational complexity of the bivariate polynomial interpolation and that of radial basis function interpolation.

As all know, every natural cubic spline $s(x)$ can be represented in the form [21]

$$s(x) = \sum_{j=1}^{N} \alpha_j \phi(|x - x_j|) + p(x), \quad x \in \mathbb{R}, \tag{6.1}$$

where $\phi(r) = r^3, r \geq 0$, and $p(x) \in \mathbb{P}_1(\mathbb{R})$, i.e., a univariate polynomial with 1 degree.

From the first result on radial basis functions [21], the resulting interpolation is up to a low-degree polynomial a linear combination of shifts of a radial function $\Phi = \phi(|\cdot|)$. Consequently, the idea has been generalized to construct scattered data interpolation of the form over the interpolating nodes $\Omega \equiv \Omega_N = \{\mathbf{x}_0, \cdots, \mathbf{x}_N\} \subset \mathbb{R}^2$

$$s(\mathbf{x}) = \sum_{j=0}^{N} \alpha_j \phi(\|\mathbf{x} - \mathbf{x}_j\|_2) + p(x), \quad \mathbf{x} = (x, y), \mathbf{x}_j = (x_j, y_j) \in \mathbb{R}^2, \tag{6.2}$$

where $\phi : [0, +\infty] \to \mathbb{R}$ is a univariate fixed function and $p(\mathbf{x}) \in \mathbb{P}_{m-1}(\mathbb{R}^2)$, i.e., a bivariate polynomial with $m-1$ degrees. And the additional conditions on the coefficients satisfy

$$\sum_{j=0}^{N} \alpha_j q(\mathbf{x}_j) = 0, \quad \forall q(\mathbf{x}) \in \mathbb{P}_{m-1}(\mathbb{R}^2). \tag{6.3}$$

Particularly, We only consider the interpolation problems without the additional conditions (6.3), which boils down to the question whether the matrix $A_{\phi,\Omega} = (\phi(\|\mathbf{x}_k -$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

$\mathbf{x}_j\|_2))_{0\le k,j\le N}$ is nonsingular. Astonishingly, there exist nonsingular matrix $A_{\phi,\Omega}$ in the case of the Gaussians $\phi(r)=e^{-\alpha r^2}(\alpha>0)$, the inverse multiquadric $\phi(r)=1/\sqrt{c^2+r^2}$, and the multiquadric $\phi(r)=\sqrt{c^2+r^2},(c>0)$.

Hence, we can obtain the α_i's by solving the system of the linear equations.

$$s(\mathbf{x}_k)=\sum_{j=0}^{N}\alpha_j\phi(\|\mathbf{x}_k-\mathbf{x}_j\|_2),\quad k=0,1,\cdots,N. \tag{6.4}$$

We make a total count of operations for the elimination step of Gaussian elimination [15]. The elimination of each entry in the matrix $A_{\phi,\Omega}$ requires the operations of addition/substractions, multiplication, and divisions, which are

$$\frac{2}{3}(N+1)^3+\frac{1}{2}(N+1)^2-\frac{7}{6}(N+1),$$

i.e., $O(N^3)$. Meanwhile, because of the triangular shape of the nonzero coefficients of the equations, we start at the bottom and work your way up to the top equation. Counting operations of solving the system of the linear equations (6.4) yields $(N+1)^2$. Consequently, counting operations to calculate $s(\mathbf{x})$ over Ω_N yields

$$\begin{aligned}&\left(\frac{2}{3}(N+1)^3+\frac{1}{2}(N+1)^2-\frac{7}{6}(N+1)\right)+(N+1)^2+\Big((N+1)+N\Big)\\ =&\frac{2}{3}(N+1)^3+\frac{3}{2}(N+1)^2+\frac{5}{6}(N+1)-1.\end{aligned} \tag{6.5}$$

Specifically, we summarize the results on the odd and even interpolating nodes.

Proposition 6.1 *The computation step for $s(\mathbf{x})$ over Ω_{2n} can be completed in*

$$RBFNUM_{2n}=\frac{2}{3}(2n+1)^3+\frac{3}{2}(2n+1)^2+\frac{5}{6}(2n+1)-1 \tag{6.6}$$

operations by means of radial basis functions, including addition/substractions, multiplication, and divisions. Meanwhile, the computation step for $s(\mathbf{x})$ over Ω_{2n+1} can be completed in

$$RBFNUM_{2n+1}=\frac{2}{3}(2n+2)^3+\frac{3}{2}(2n+2)^2+\frac{5}{6}(2n+2)-1 \tag{6.7}$$

operations.

Then we shall count the number of operations with the bivariate polynomial interpolation presented.

Proposition 6.2 *The computation step for $p_{2n}(x,y)$ in* (2.1) *over Ω_{2n} can be completed in*

$$PINUM_{2n}=2n^3+\frac{15}{2}n^2+\frac{29}{2}n-4 \tag{6.8}$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

operations, including addition/substractions, multiplication, and divisions. Meanwhile, the computation step for $p_{2n+1}(x,y)$ *in* (2.2) *over* Ω_{2n+1} *can be completed in*

$$PINUM_{2n+1} = 2n^3 + \frac{21}{2}n^2 + \frac{47}{2}n + 4 \tag{6.9}$$

operations, including addition/substractions, multiplication, and divisions.

Actually, we analyze the computational complexity of calculating the bivariate interpolating polynomial $p_{2n}(x,y)$ in (2.1) over Ω_{2n} from two aspects firstly.

On one hand, by means of Definition 2.1 and Algorithm 2.1, calculating the interpolating coefficients a_1 up to a_{2n} requires the operation of addition/substractions

$$\begin{aligned} &2\times(2n) + \sum_{k=1}^{n}\big(2(2n+1-2k) + (2k-1)(n-k+1)\big) + \sum_{k=1}^{n-1}\big(2\times(2n-2k) + 2k(n-k)\big) \\ &= \frac{2}{3}n^3 + \frac{9}{2}n^2 + \frac{11}{6}n, \end{aligned}$$

the operation of multiplication

$$\sum_{k=1}^{n}\Big(2(2k-2)+1\Big)(n-k+1) + \sum_{k=1}^{n-1}\Big(2(2k-1)+1\Big)(n-k) = \frac{4}{3}n^3 - \frac{1}{3}n,$$

and the operation of divisions

$$2n + \sum_{k=1}^{n}\Big(2(n-k+1) + (n-k)\Big) + \sum_{k=1}^{n-1}\Big(2(n-k) + (n-k)\Big) = 3n^2 + n,$$

which is totaled up as $2n^3 + 7.5n^2 + 2.5n$.

On the other hand, by (2.1), calculating the bivariate interpolating polynomial $p_{2n}\ (x,y)$ with the known coefficients additionally requires the operation of addition/substractions

$$2n + 1 + 2(2n-1) = 6n - 1,$$

the operation of multiplication $1 + 2 + 3(2n-2) = 6n - 3$.

Hence, we total up the number as

$$2n^3 + \frac{15}{2}n^2 + \frac{29}{2}n - 4,$$

which gives (6.8).

Secondly, we consider the computational complexity of calculating the bivariate interpolating polynomial $p_{2n+1}(x,y)$ in (2.2) over Ω_{2n+1} from two aspects.

On one hand, by means of Definition 2.1 and Algorithm 2.2, the calculation step for the interpolating coefficients a_1 up to a_{2n+1} can be completed in the addition/substractions

$$\left(\frac{2}{3}n^3 + \frac{9}{2}n^2 + \frac{11}{6}n\right) + 2(2n+1) + \sum_{k=1}^{n}2k = \frac{2}{3}n^3 + \frac{11}{2}n^2 + \frac{41}{6}n + 2,$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

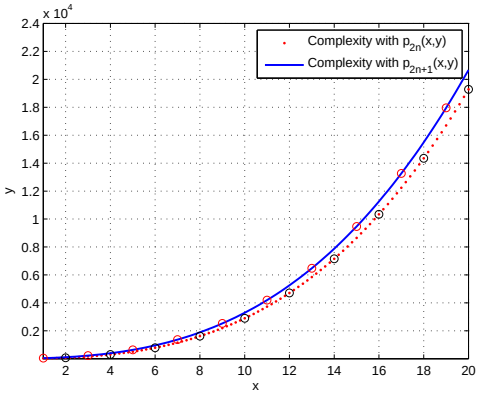


Figure 13: The computational complexity with $p_{2n}(x,y)(n=1,2,\cdots,10)$ and $p_{2n+1}(x,y)(n=0,1,\cdots,9)$.

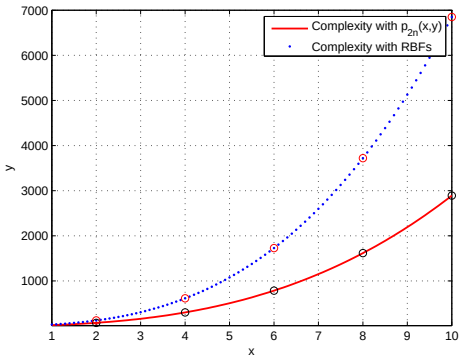


Figure 14: The computational complexity with $p_{2n}(x,y)$ $(n=1,2,3,4,5)$ and RBFs.

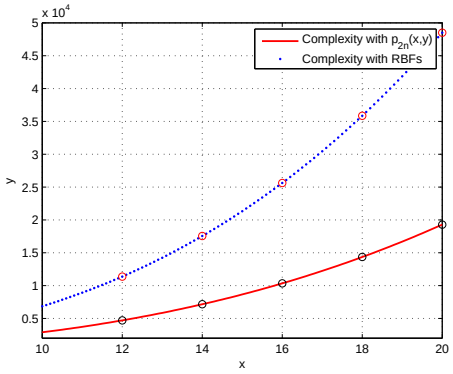


Figure 15: The computational complexity with $p_{2n}(x,y)$ $(n=6,7,8,9,10)$ and RBFs.

the multiplication

$$\left(\frac{4}{3}n^3-\frac{1}{3}n\right)+\sum_{k=1}^{n}(4k-1)=\frac{4}{3}n^3+2n^2+\frac{2}{3}n,$$

the divisions

$$(3n^2+n)+(1+n+2n)=3n^2+4n+1=3n^2+n$$

operations, which is totaled up as $2n^3+10.5n^2+11.5n+3$.

On the other hand, by (2.2), the calculation step for the bivariate interpolating polynomial $p_{2n+1}(x,y)$ with the known coefficients can additionally be completed in the addition/substractions $(6n-1)+2=6n+1$, and the operation of multiplication $(6n-3)+3=6n$ operations.

Finally, we total up the number as

$$2n^3+\frac{21}{2}n^2+\frac{47}{2}n+4,$$

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

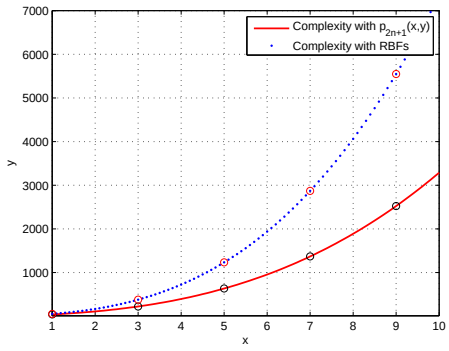


Figure 16: The computational complexity with $p_{2n+1}(x, y)$ $(n = 0, 1, 2, 3, 4)$ and RBFs.

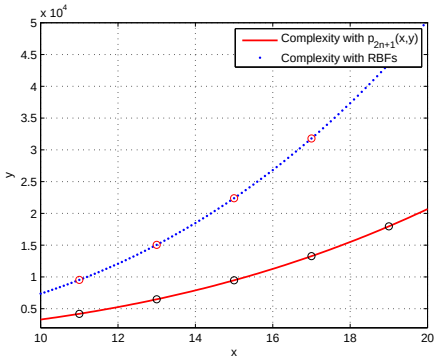


Figure 17: The computational complexity with $p_{2n+1}(x, y)$ $(n = 5, 6, 7, 8, 9)$ and RBFs.

which gives (6.9). As a result, it follows that Proposition 6.2 holds.

For the sake of illustration, we plot the figures of the number of operations in Propositions 6.1 and 6.2 to compare the computational complexity of the bivariate polynomial presented with that of radial basis function, as shown in Figs. 13-17. Moreover, we obtain

Proposition 6.3 *For the computation of the interpolating functions over Ω_{2n} or Ω_{2n+1} based on the bivariate interpolating polynomials presented and radial basis functions, the quotient of the number of the operations of addition/substractions, multiplication, and divisions satisfies*

$$\frac{PINUM}{RBFNUM} \approx \frac{3}{8}$$

for sufficiently large n.

Acknowledgments This work was supported by special foundation Tianyuan of the National Natural Science Foundation of China (Nos. 11426086), the Fundamental Research Funds for the Central Universities (Nos. 2016B08714), and the Natural Science Foundation of Jiangsu Province for the Youth (Grant No. BK20160853).The authors appreciate the comments and valuable suggestions of the editor and anonymous reviewers.

References

[1] G. Allasia, and C. Bracco, *Multivariate Hermite-Birkhoff interpolation by a class of cardinal basis function*, Appl. Math. Comput., vol. 218 (2012), pp. 9248–9260.

[2] B. A. Bailey, *Multivariate polynomial interpolation and sampling in Paley-Wiener spaces*, J. Approx. Theory, vol. 164 (2012), pp. 460–487.

[3] J. Chai, N. Lei, Y. Li, and P. Xia, *The proper interpolation space for multivariate Birkhoff interpolation*, J. Comput. Appl. Math., vol. 235 (2011), pp. 3207–3214.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press

[4] K. Cui, and N. Lei, *Stable monomial basis for multivariate Birkhoff interpolation problems*, J. Comput. Appl. Math., vol. 277 (2015), pp. 162–170.
[5] N. Dyn, and M. S. Floater, *Multivariate polynomial interpolation on lower sets*, J. Approx. Theory, vol. 177 (2014), pp. 34–42.
[6] M. Gasca, T. Sauer, *On the history of multivariate polynomial interpolation*, J. Comput. and Appl. Math., vol. 122 (2000), pp. 23–35.
[7] M. J. Lai, *Convex preserving scattered data interpolation using bivariate C^1 cubic splines*, J. Comput. Appl. Math., vol. 119 (2000), pp. 249–258.
[8] C. J. Li, and R. H. Wang,, *Bivariate cubic spline space and bivariate cubic NURBS surfaces*, Proceedings of Geometric Modeling and Processing 2004, IEEE Computer Society Pressvol, pp. 115–123.
[9] W. R. Madych, *An estimate for multivariate interpolation II*, J. Approx. Theory, vol. 142 (2006), pp. 116–128.
[10] A. Mazroui, D. Sbibih, and A. Tijini, *Recursive computation of bivariate Hermite spline interpolants*, Appl. Numer. Math., vol. 57 (2007), pp. 962–973.
[11] J. Qian, F. Wang, *On the approximation of the derivatives of spline quasi-interpolation in cubic spline space $S_3^{1,2}(\Delta_{mn}^{(2)})$*, Numer. Math. Theor. Meth. Appl. vol., 7(1) (2014), pp. 1–22.
[12] J. Qian, R. H. Wang, C. J. Li, *The bases of the Non-uniform cubic spline space $S_3^{1,2}(\Delta_{mn}^{(2)})$*, Numer. Math. Theor. Meth. Appl. vol. 5(4) (2012), pp. 635–652.
[13] J. Qian, R. H. Wang, C. G. Zhu, F. Wang, *On spline quasi-interpolation in cubic spline space $S_3^{1,2}(\Delta_{mn}^{(2)})$*, Sci Sin Math, vol. 44(7) (2014), pp. 769–778, (in Chinese).
[14] H.E. Salzer, *Some new divided difference algorithm for two variables*, in: R.E. Langer (Ed.) On Numerical Approximation, 1959.
[15] T. Sauer, *Numerical Analysis*, China Machine Press, Beijing, 2012.
[16] R. H. Wang, *Numerical Approximation*, Beijing, High Education Press, 1999.
[17] R. H. Wang, and C. J. Li, *Bivariate quartic spline spaces and quasi-interpolation operators*, J. Comput. and Appl. Math., vol. 190, (2006), pp. 325–338.
[18] R. H. Wang, J. Qian, *On branched continued fractions rational interpolation over pyramid-typed grids*, Numer. Algor., vol. 54 (2010), pp. 47–72.
[19] R. H. Wang, J. Qian, *Bivariate polynomial and continued fraction interpolation over ortho-triples*, Applied Mathematics and Computation, vol. 217 (2011), pp. 7620–7635.
[20] R. H. Wang, X. Q. Shi, Z. X. Luo, and Z. X. Su, *Multivariate Spline Functions and Their Applications*, Science Press/Kluwer Academic Publishers, Beijing, New York, Dordrecht, Boston, London, 2001.
[21] H. Wendland, *Scattered Data Approximation*, Cambridge University Press, 2005.
[22] T. H. Zhou, and M. J. Lai, *Scattered data interpolation by bivariate splines with higher approximation order*, J. Comput. Appl. Math., vol. 242 (2013), pp. 125–140.
[23] C. G. Zhu, and R. H. Wang, *Lagrange interpolation by bivariate splines on cross-cut partitions*, J. Comp. Appl. Math., vol. 195 (2006), pp. 326–340.

https://doi.org/10.4208/nmtma.2016.y15027 Published online by Cambridge University Press